In the claims:

1. A computer-implemented method for substantially eliminating C recursion from the execution of static initializer methods in a virtual machine environment, the method comprising:

rewriting native C code associated with a static initializer as a Java™ method;

using a transition frame in a Java™ stack to execute the Java™ method;

using a native method to manipulate the Java™ stack; and

using a first opcode in the transition frame.

2. A method as recited in claim 1 wherein using the first opcode in the transition frame includes using the first opcode to determine that the transition frame is associated with the static initializer.

3. A method as recited in claim 2 further including causing the static initializer to run, wherein the static initializer is caused to run by a second opcode.

4. A method as recited in claim 3 further including resuming execution at the second opcode after the static initializer has run.

5. A method as recited in claim 1wherein using the native method enables the static initializer to execute without re-entering an interpreter.

6. A method as recited in claim 1 wherein the native C code includes code for identifying the static initializer.

7. An apparatus for substantially eliminating C recursion from the execution of static initializer methods in a virtual machine environment, the method comprising:

a means for rewriting native C code associated with a static initializer as a Java™ method;

a means for using a transition frame in a Java™ stack to execute the Java™ method;

a means for using a native method to manipulate the Java™ stack; and

a means for using a first opcode in the transition frame.

8. An apparatus as recited in claim 7 further comprising:

a means for using the first opcode to determine that the transition frame is associated with the static initializer.

9. An apparatus as recited in claim 8 further comprising:

a means for causing the static initializer to run, wherein the static initializer is caused to run by a second opcode.

10. An apparatus as recited in claim 9 further comprising:

a means for resuming execution at the second opcode after the static initializer has run.

11. An apparatus as recited in claim 7 wherein the native C code includes code for identifying the static initializer.